LLNL-JRNL-408814

Leveraging Label-Independent Features for Classification in Sparsely Labeled Networks: An Empirical Study

B. Gallagher, T. Eliassi-Rad

November 18, 2008

Springer Lecture Notes in Computer Science

Disclaimer

Leveraging Label-Independent Features for Classification in Sparsely Labeled Networks: An Empirical Study

Brian Gallagher and Tina Eliassi-Rad

Lawrence Livermore National Laboratory
P.O. Box 808, L-560, Livermore, CA 94551, USA
{bgallagher, eliassi}@llnl.gov

Abstract. We address the problem of *within-network classification* in sparsely labeled networks. Recent work has demonstrated success with *statistical relational learning* (SRL) and *semi-supervised learning* (SSL) on such problems. However, both approaches rely on the availability of labeled nodes to infer the values of missing labels. When few labels are available, the performance of these approaches can degrade. In addition, many such approaches are sensitive to the specific set of nodes labeled. So, although average performance may be acceptable, the performance on a specific task may not. We explore a complimentary approach to within-network classification, based on the use of *label-independent* (*LI*) *features* – i.e., features calculated without using the values of class labels. While previous work has made some use of LI features, the effects of these features on classification performance have not been extensively studied. Here, we present an empirical study in order to better understand these effects. Through experiments on several real-world data sets, we show that the use of LI features produces classifiers that are less sensitive to specific label assignments and can lead to performance improvements of over 40% for both SRL- and SSL-based classifiers. We also examine the relative utility of individual LI features; and show that, in many cases, it is a combination of a few diverse network-based structural characteristics that is most informative.

Keywords: Statistical relational learning; semi-supervised learning; social network analysis; feature extraction; collective classification.

1 Introduction

In this paper, we address the problem of within-network classification. We are given a network in which some of the nodes are "labeled" and others are "unlabeled" (see Figure 1). Our goal is to assign the correct labels to the unlabeled nodes from among a set of possible class labels (i.e., to "classify" them). For example, we may wish to identify cell phone users as either 'fraudulent' or 'legitimate.'

Cell phone fraud is an example of an application where networks are often very sparsely labeled. We may have a handful of known fraudsters and a handful

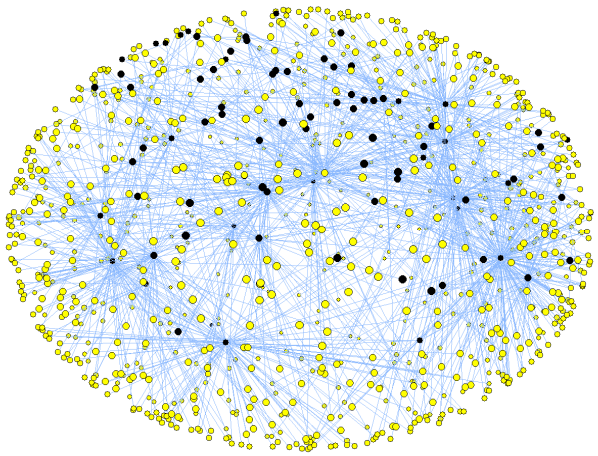

Fig. 1. Portion of the MIT Reality Mining call graph. We know the class labels for the black (dark) nodes, but do not have labels for the yellow (light) nodes.

of known legitimate users, but for the vast majority of users, we do not know the correct label. For such applications, it is reasonable to expect that we may have access to labels for fewer than 10%, 5%, or even 1% of the nodes. In addition, cell phone networks are generally anonymized. That is, nodes in these networks often contain no attributes besides class labels that could be used to identify them. It is this kind of sparsely labeled, anonymized network that is the focus of this work. Put another way, our work focuses on *univariate within-network classification in sparsely labeled networks.*

Relational classifiers have been shown to perform well on network classification tasks because of their ability to make use of dependencies between class labels (or attributes) of related nodes [1]. However, because of their dependence on class labels, the performance of relational classifiers can substantially degrade when a large proportion of neighboring instances are also unlabeled. In many cases, *collective classification* provides a solution to this problem, by enabling the simultaneous classification of a number of related instances [2]. However, previous work has shown that the performance of collective classification can also degrade when there are too few labels available, eventually to the point where classifiers perform better without it [3].

In this paper, we explore another source of information present in networks that does not depend on the availability or accuracy of node labels. Such information can be represented using what we call *label-independent* (*LI*) *features.* The main contribution of this paper is an in-depth examination of the effects of label-independent features on within-network classification. In particular, we address the following questions:

1. *Can LI features make up for a lack of information due to sparsely labeled data*? Answer: Yes.

2. *Can LI features provide information above and beyond that provided by the class labels*? Answer: Yes.

3. *How do LI features improve classification performance*? Answer: Because they are less sensitive to the specific labeling assigned to a graph, classifiers that use label-independent features produce more consistent results across prediction tasks.

4. *Which LI features are the most useful*? Answer: A combination of a few diverse network-based structural characteristics (such as node and link counts plus betweenness) is the most informative.

Section 2 covers related work. Section 3 describes our approach for modeling label-independent characteristics of networks. Sections 4 and 5, respectively, present our experimental design and results. We conclude the paper in Section 6.

2 Related Work

In recent years, there has been a great deal of work on models for learning and inference in relational data (i.e., statistical relational learning or SRL) [3–7]. All SRL techniques make use of label-dependent relational information. Some use label-independent information as well.

Relational Probability Trees (RPTs) [8] use label-independent degree-based features (i.e., neighboring node and link counts). However, existing RPT studies do not specifically consider the impact of label-independent features on classifier performance.

Perlich and Provost [9] provide a nice study on aggregation of relational attributes, based on a hierarchy of relational concepts. However, they do not consider label-independent features.

Singh et al. [10] use descriptive attributes and structural properties (i.e., node degree and betweenness centrality) to prune a network down to its 'most informative' affiliations and relationships for the task of attribute prediction. They do not use label-independent features directly as input to their classifiers.

Neville and Jensen [11] use spectral clustering to group instances based on their link structure (where link density within a group is high and between groups is low). This group information is subsequently used in conjunction with attribute information to learn classifiers on network data.

There has also been extensive work on overcoming label sparsity through techniques for label propagation. This work falls into two research areas: (1) collective classification [2, 3, 7, 12–14] and (2) graph-based semi-supervised learning (SSL) [15, 16].

Previous work confirms our observation that the performance of collective classification can suffer when labeled data is very sparse [3]. McDowell et al. [14]

demonstrate that "cautious" collective classification procedures produce better classification performance than "aggressive" ones. They recommend only propagating information about the top-k most confident predicted labels.

The problem of within-network classification can be viewed as a semi-supervised learning problem. The graph-based approaches to semi-supervised learning are particularly relevant here. In their study, Macskassy and Provost [7] compare the SSL Gaussian Random Field (GRF) model [15] to a SRL weighted-vote relational neighbor (wvRN) model that uses relaxation labeling for collective classification (wvRN+RL). They conclude that the two models are nearly identical in terms of accuracy, although GRF produces slightly better probability rankings. Our results with wvRN+RL and GRF are consistent with this conclusion. The "ghost edge" approach of Gallagher et al. [17] combines aspects of both SRL and SSL, and compares favorably with both wvRN+RL and GRF.

3 Label-Dependent vs. Label-Independent Features

Relational classifiers leverage link structure to improve performance. Most frequently, links are used to incorporate attribute information from neighboring nodes. However, link structure can also be used to extract structural statistics of a node (e.g., the number of adjacent links). We can divide relational features into two categories: label-dependent and label-independent.

Label-dependent (LD) features use both structure and attributes (or labels) of nodes in the network. The most commonly used LD features are aggregations of the class labels of nodes one link away (e.g., the number of neighbors with the class label 'fraudulent'). LD features are the basis for incorporating relational information in many SRL classifiers.

Label-independent (LI) features are calculated using network structure, but not attributes or class labels of nodes. An example of a simple LI feature is the degree of a node (i.e., the number of neighboring nodes). Of course, we assume that there is an underlying statistical dependency between the class label of a node and its LI features. Otherwise, LI features would be of no value in predicting a node's class. However, because they are calculated based only on network structure, LI feature values do not directly depend on the current class label assignments of nodes in a network. This means that, unlike LD features, LI features may be calculated with perfect accuracy regardless of the availability of class label information and are impervious to errors in class label assignments.

3.1 Extracting Label-Independent Features

We consider four LI features on nodes: (1) the number of neighboring nodes, (2) the number of incident links, (3) betweenness centrality, and (4) clustering coefficient. Features 1 and 2, respectively, are node-based and link-based measures of degree. Note that in multigraphs, these two are different. Betweenness centrality measures how "central" a node is in a network, based on the number

of shortest paths that pass through it. Clustering coefficient measures neighborhood strength, based on how connected a node's neighbors are to one another. For details, we refer the reader to a study by Mark Newman [18].

The success of network-based structural characteristics as predictors of class relies on two assumptions. First, members of different classes play different roles in a network. Second, these roles can be differentiated by structural characteristics. The second assumption is met in many cases. For instance, "popular" nodes can be identified by degree and "important" nodes can be identified by centrality measures. Whether the first assumption is met depends on the class label. Suppose that executives tend to be more popular and central than an average employee in a company's communication network, and that employees with a particular job title tend to have similar popularity and centrality, regardless of department. Then, we would expect structural features to be more useful for identifying executives than members of a particular department.

4 Experimental Design

We have designed our experiments to answer the following questions:

1. *Can LI features make up for a lack of information due to sparsely labeled data*?

2. *Can LI features provide information above and beyond that provided by the class labels*?

3. *How do LI features improve classification performance*?

4. *Which LI features are the most useful*?

To avoid confounding effects as much as possible, we focus on univariate binary classification tasks, and extend simple classifiers to incorporate label-independent features.

4.1 Classifiers

On each classification task, we ran ten individual classifiers: four variations of a link-based classifier [5], four variations of a relational neighbor classifier [19, 7], and two variations of the Gaussian Random Field classifier [15]. We describe each of them below.

nLB is the network-only link-based classifier [5]. It uses logistic regression to model a node's class given the classes of neighboring nodes. To generate features, a node's neighborhood is summarized by the *link-weighted count* of each class label. For example, given a binary classification task, two features will be generated: (1) the link-weighted count of a node's neighbors with the positive

class and (2) the link-weighted count of a node's neighbors with the negative class.

nLBLI is composed of two logistic regression models: (1) nLB with its two LD features and (2) logLI, which uses the four LI features (see Section 3.1). The nLBLI classifier calculates the probability of each class as:

$$P(C) = w \cdot P_{nLB}(C) + (1 - w) \cdot P_{logLI}(C) \tag{1}$$

where w is calculated based on the individual performance of nLB and logLI over 10-fold cross validation on the training data. We calculate area under the ROC curve (AUC) for each fold and then obtain an average AUC score for each classifier, AUC_{LD} and AUC_{LI}. We then set w as follows:

$$w = \frac{AUC_{LD}}{AUC_{LD} + AUC_{LI}} \tag{2}$$

nLB+ICA uses the nLB classifier, but performs collective classification using the ICA algorithm described in Section 4.2.

nLBLI+ICA uses the nLBLI classifier, but performs collective classification using the ICA algorithm described in Section 4.2.

wvRN is the weighted-vote relational neighbor classifier [19, 7]. It is a simple non-learning classifier. Given a node i and a set of neighboring nodes, N, the wvRN classifier calculates the probability of each class for node i as:

$$P(C_i = c|N) = \frac{1}{L_i} \sum_{j \in N} \begin{cases} w_{i,j} & \text{if } C_i = c \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $w_{i,j}$ is the number of links between nodes i and j and L_j is the number of links connecting node i to labeled nodes. When node i has no labeled neighbors, we use the prior probabilities observed in the training data.

wvRNLI combines the LI features with wvRN in the same way that nLBLI does with nLB (i.e., using a weighted sum of wvRN and logLI).

wvRN+ICA uses the wvRN classifier, but performs collective classification using the ICA algorithm described in Section 4.2.

wvRNLI+ICA uses wvRNLI, but performs collective classification using the ICA algorithm described in Section 4.2.

GRF is the semi-supervised Gaussian Random Field approach of Zhu et al. [15]. We made one modification to accommodate disconnected graphs. Zhu computes the graph Laplacian as $L = D - cW$, where $c = 1$. We set $c = 0.9$ to ensure

that L is diagonally dominant and thus invertible. We observed no substantial impact on performance in connected graphs due to this change.

GRFLI combines the LI features with GRF as nLBLI does with nLB (i.e., using a weighted sum of GRF and logLI). We also tried the approach of Zhu et al. [15], where one attaches a "dongle" node to each unlabeled node and assigns it a label using the external LI classifier. The transition probability from node i to its dongle is η and all other transitions from i are discounted by $1 - \eta$. This approach did not yield any improvements. So, we use the weighted sum approach (i.e., Equation 1) for consistency.

4.2 Collective Classification

To perform collective classification, we use the iterative classification algorithm (ICA) [7], with up to 1000 iterations. We chose ICA because (1) it is simple, (2) it performs well on a variety of tasks, and (3) it tends to converge more quickly than other approaches. We also performed experiments using relaxation labeling (RL) [7]. Our results are consistent with previous research showing that the accuracy of wvRN+RL is nearly identical to GRF, but GRF produces higher AUC values [7]. We omit these results due to the similarity to GRF. For a comparison of wvRN+RL and GRF on several of the same tasks used here, see Gallagher et al. [17]. Overall, ICA slightly outperforms RL for the nLB classifier.

Several of our data sets have large amounts of unlabeled data since ground truth is simply not available. In these cases, there are two reasonable approaches to collective classification: (1) perform collective classification over the entire graph and (2) perform collective classification over the core set of nodes only (i.e., nodes with known labels).

In our experiments, attempting to perform collective classification over the entire graph produced results that were often dramatically worse than the non-collective base classifier. We hypothesize that this is due to an inadequate propagation of known labels across vast areas of unlabeled nodes in the network. Note that for some of our experiments, fewer than 1% of nodes are labeled. Other researchers have also reported cases where collective classification hurts performance due to a lack of labeled data [3, 11]. We found that the second approach (i.e., using a network of only the core nodes) outperformed the first approach in almost all cases, despite disconnecting the network in some cases. Therefore, we report results for the second approach only.

4.3 Experimental Methodology

Each data set has a set of core nodes for which we know the true class labels. Several data sets have additional nodes for which there is no ground truth available. Classifiers have access to the entire graph for both training and testing. However, we hide labels for $10\% - 90\%$ of the core nodes. Classifiers are trained on all labeled core nodes and evaluated on all unlabeled core nodes.

For each proportion labeled, we run 30 trials. For each trial, we choose a class-stratified random sample containing $100 \times (1.0 - proportionlabeled)\%$ of the core nodes as a test set and the remaining core nodes as a training set. Note that a single node will necessarily appear in multiple test sets. However, we carefully choose test sets to ensure that each node in a data set occurs in the same number of test sets over the course of our experiments; and therefore, carries the same weight in the overall evaluation. Labels are kept on training nodes and removed from test nodes. We use identical train/test splits for each classifier. For more on experimental methodologies for relational classification, see Gallagher and Eliassi-Rad [20].

We use the area under the ROC curve (AUC) to compare classifiers because it is more discriminating than accuracy. In particular, since most of our tasks have a large class imbalance (see Section 4.4), accuracy cannot adequately differentiate between classifiers.

4.4 Data Sets

We present results on four real-world data sets: political book purchases [21], Enron emails [22], Reality Mining (RM) cellphone calls [23], and high energy physics publications (HEP-TH) from arXiv [24]. Our five tasks are to identify neutral political books, Enron executives, Reality Mining students, Reality Mining study participants, and HEP-TH papers with the topic "Differential Geometry." Table 1 summarizes the prediction tasks. The *Sample* column describes the method used to obtain a data sample for our experiments: use the entire set (*full*), use a time-slice (*time*), or sample a continuous subgraph via breadth-first search (*BFS*). The *Task* column indicates the class label we try to predict. The $|V|$, $|L|$, and $|E|$ columns indicate counts of total nodes, labeled nodes, and total edges in each network. The $P(+)$ column indicates the proportion of labeled nodes that have the positive class label (e.g., 12% of the political books are neutral). For Enron, Reality Mining students, and HEP-TH, we have labels for only a subset of nodes (i.e., the "core" nodes) and can only train and test our classifiers on these nodes. However, unlabeled nodes and their connections to labeled nodes are exploited to calculate LI features of the labeled nodes.

Table 1. Summary of Data Sets and Prediction Tasks.

Data Set	Sample	Task	$\|V\|$	$\|L\|$	$\|E\|$	$P(+)$
Political Books	Full	Neutral?	105	105	441	0.12
Enron	Time	Executive?	9K	1.6K	50K	0.02
Reality Mining	BFS	Student?	1K	84	32K	0.62
Reality Mining	BFS	In Study?	1K	1K	32K	0.08
HEP-TH	BFS	Differential Geometry?	3K	284	36K	0.06

5 Experimental Results

In this section, we discuss our results. We assess significance using paired t-tests (p-values ≤ 0.05 are considered significant).[1]

5.1 Effects of Learning Label Dependencies

Figures 2 and 3 show results for statistical relational learning and semi-supervised learning approaches on all of our classification tasks. Supervised learning approaches, like nLB, use labeled nodes as training data to build a dependency model over neighboring class labels. The non-learning wvRN and GRF assume that class labels of neighboring nodes tend to be the same (i.e., high label consistency). GRF performs well on the Enron and RM student tasks, which have high label consistency between neighbors. On the RM study task, where neighboring labels are inversely correlated (i.e., low label consistency), wvRN and GRF perform poorly, whereas nLB can learn the correct dependencies.

5.2 Effects of Label-Independent Features

Figures 2 and 3 illustrate several effects of LI features. In general, the performance of the LI classifiers degrades more slowly than that of the corresponding base classifiers as fewer nodes are labeled. At $\leq 50\%$ labeled, the LI features produce a significant improvement in 36 of 45 cases. The exceptions mainly occur for GRF on Enron, RM Student, and HEP-TH, where (in most cases) we have a statistical tie. In general, the information provided by the LI features is able to make up, at least in part, for information lost due to missing labels. Note that there are three separate effects that lower performance as the number of labels decreases. (1) Fewer labels available for inference lead to lower quality LD features at inference time, but do not impact the quality of LI features. (2) Fewer labels at training time mean that (labeled) training examples have fewer labeled neighbors. This impacts the quality of the LD features available at training time and the quality of the resulting model. LI features are not affected. (3) Fewer labels mean less training data. This impacts model quality for both LD and LI features. Note that wvRN and GRF are affected only by 1, since they do not rely on training data.

In general, the LI models outperform the corresponding base models, leading to significant improvements in 49 out of 75 cases across all proportions of labeled data. There is only one case where the use of LI features significantly degrades performance: using GRF on the Enron task at 0.3 labeled. The GRF classifier

[1] It is an open issue whether the standard significance tests for comparing classifiers (e.g., t-tests, Wilcoxon signed-rank) are applicable for within-network classification, where there is typically some overlap in test sets across trials. It remains to be seen whether the use of such tests produces a bias and the extent of any errors caused by such a bias. This is an important area for future study that will potentially affect a number of published results.

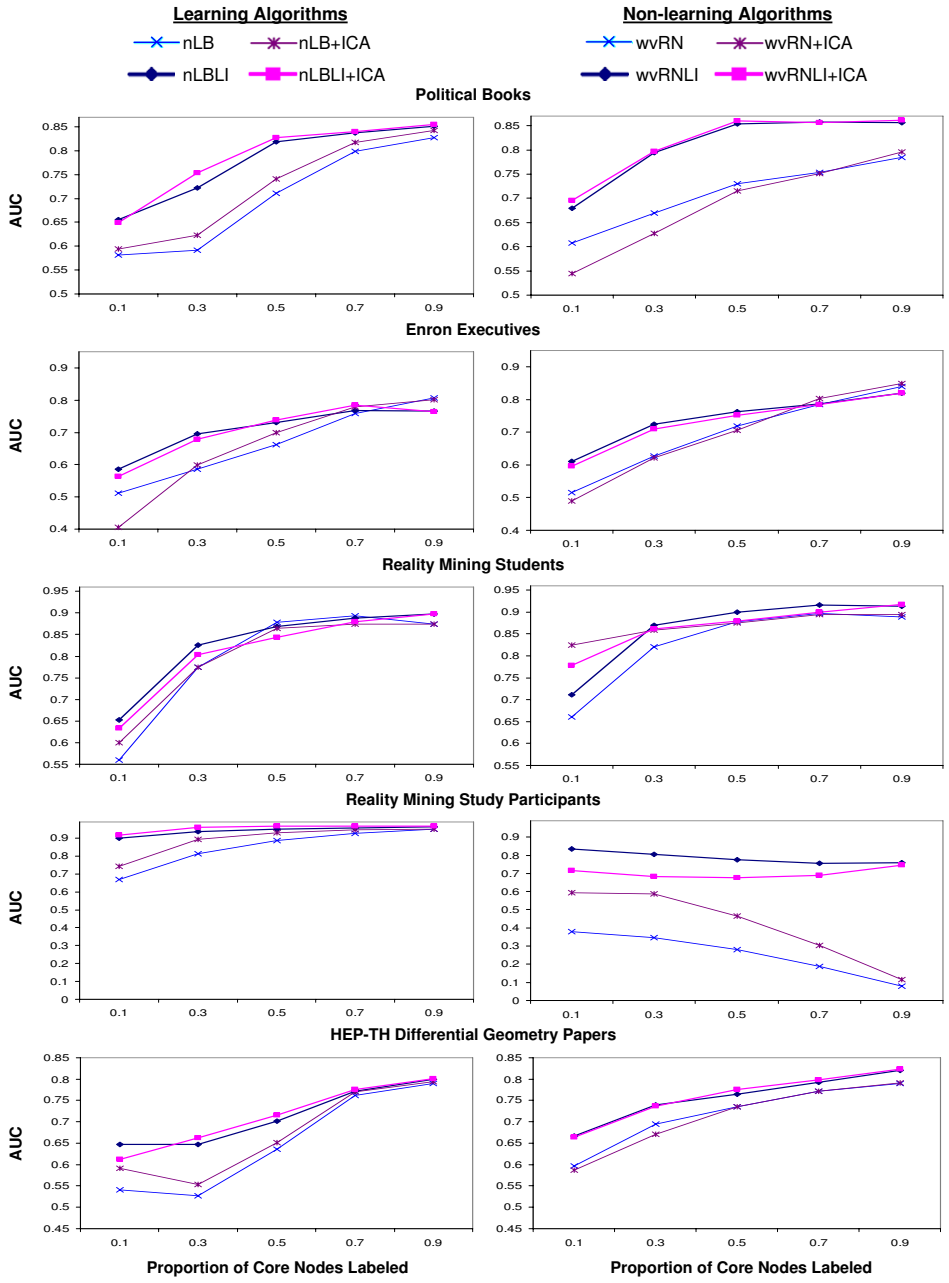

Fig. 2. Classification results for statistical relational learning approaches on our data sets. For details on classifiers, see Section 4.1. Note: Due to differences in the difficulty of classification tasks, the y-axis scales are not consistent across tasks. However, for a particular classification task, the y-axis scales are consistent across the algorithms shown both in this figure and in Figure 3.

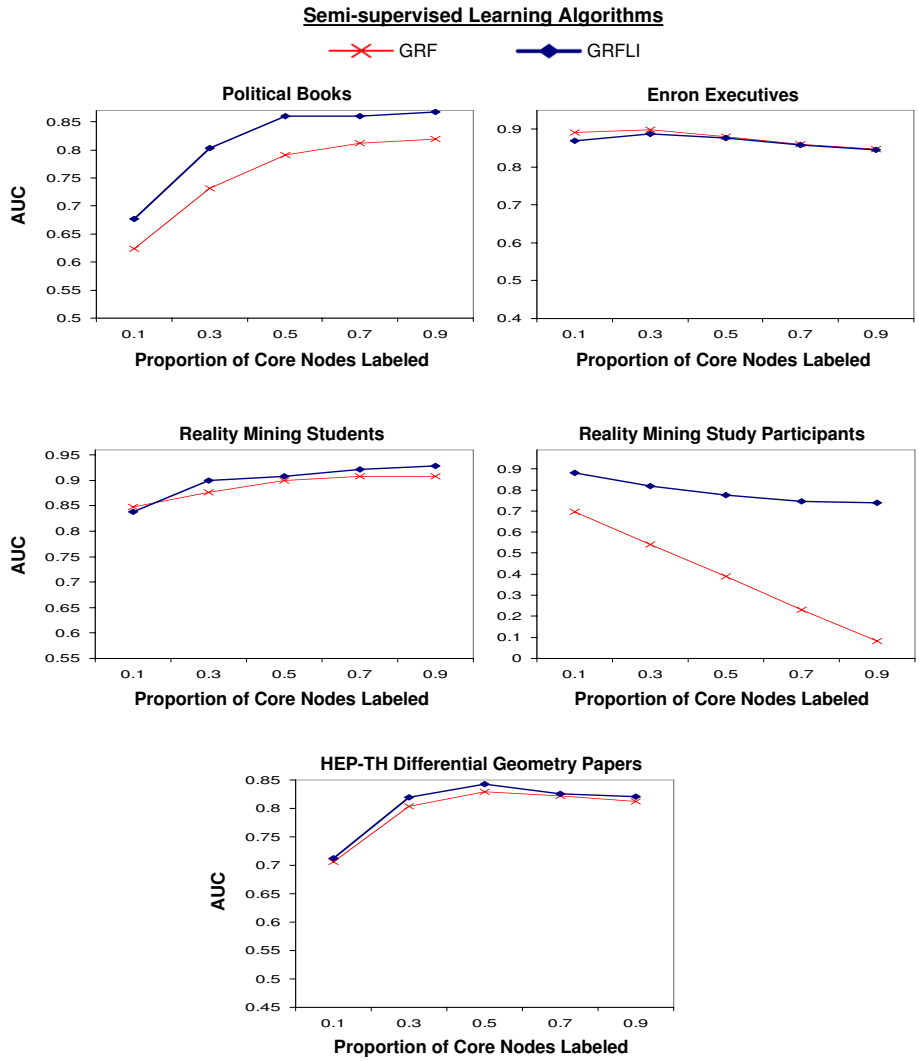

Fig. 3. Classification results for semi-supervised learning approaches on our data sets. For details on classifiers, see Section 4.1. Note: Due to differences in the difficulty of classification tasks, the y-axis scales are not consistent across tasks. However, for a particular classification task, the y-axis scales are consistent across the algorithms shown both in this figure and in Figure 2.

does so well on this task that the LI features simply add complexity without additional predictive information. However, the degradation here is small compared to gains on other tasks.

Another effect demonstrated in Figures 2 and 3 is the interaction between LI features and label propagation (i.e., ICA or GRF). In several cases, combining the two significantly outperforms either on its own (e.g., GRFLI on political books and the RM tasks). However, the benefit is not consistent across all tasks.

The improved performance due to LI features on several tasks at 90% labeled (i.e., political books, both RM tasks) suggests that LI features can provide information above and beyond that provided by class labels. Recall that political books and RM study are the only data sets fully labeled to begin with. This indicates that LI features may have more general applicability beyond sparsely labeled data.

Figure 4 shows the sensitivity of classifiers to the specific nodes that are initially labeled. For each classifier and task, we measure variance in AUC across 30 trials. For each trial, a different 50% of nodes is labeled. ICA has very little impact on the sensitivity of nLB to labeling changes. However, the LI features decrease the labeling sensitivity of nLB dramatically for all but one data set. The results for wvRN are qualitatively similar. LI features also decrease sensitivity for GRF in most cases. Since GRF has low sensitivity to begin with, the improvements are less dramatic. The observed reduction in label sensitivity is not surprising since LI features do not rely on class labels. However, it suggests that LI features make classifiers more stable. So, even in cases where average classifier performance does not increase, we expect an increase in the worst case due to the use of LI features.

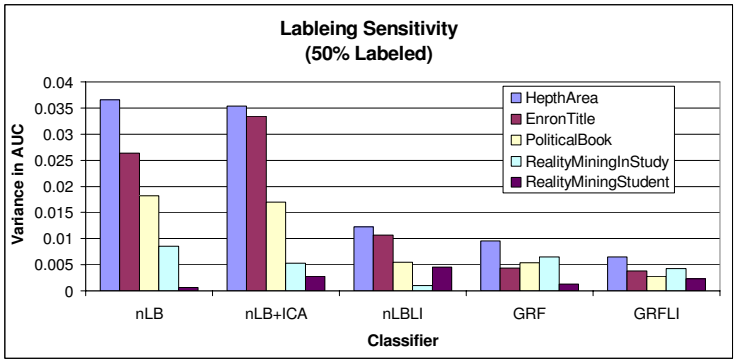

Fig. 4. Sensitivity of classifiers to specific assignments of 50% known labels across data sets.

5.3 Performance of Specific LI Features

To understand which LI features contribute to the observed performance gains, we re-ran our experiments using subsets of the LI features. We used logistic regression with different combinations of the four LI features: each alone (4 classifiers), leave one feature out (4 classifiers), degree-based features only (1 classifier), non-degree-based features only (1 classifier), and all features (1 classifier).[2] This yields 11 classifiers. We present results for 50% of nodes labeled. Results for other proportions labeled are similar.

Figure 5 shows AUC using each LI feature alone vs. all features together. This demonstrates the utility of each feature in the absence of any other information.

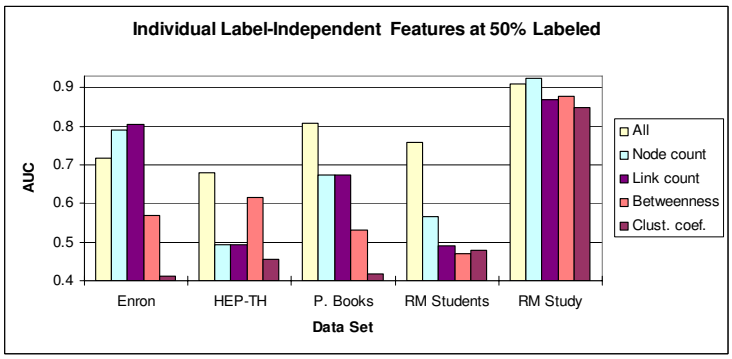

Fig. 5. Performance of LI features in isolation.

Figure 6 shows the increase in AUC due to adding the specified feature to a classifier that already has access to all other LI features. The y-axis is the AUC of a classifier that uses all LI features minus the AUC of a classifier that uses all except the specified feature. This demonstrates the power of each feature when combined with the others.

All features appear to be useful for some tasks. Clustering coefficient is the least useful overall, improving AUC slightly on two tasks and degrading AUC slightly on three. For all tasks, a combination of at least three features yields the best results. Interestingly, features that perform poorly on their own can be combined to produce good results. On the RM student task, node count, betweenness, and clustering coefficient produce AUCs of 0.57, 0.49, and 0.48 alone, respectively. When combined, these three produce an AUC of 0.78. Betweenness, which performs worse than random (AUC < 0.5) on its own, provides a boost of 0.32 AUC to a classifier using node count and clustering coefficient.

[2] Degree-based features are node (or neighbor) count and link (or edge) counts. Non-degree-based features are betweenness and clustering coefficient.

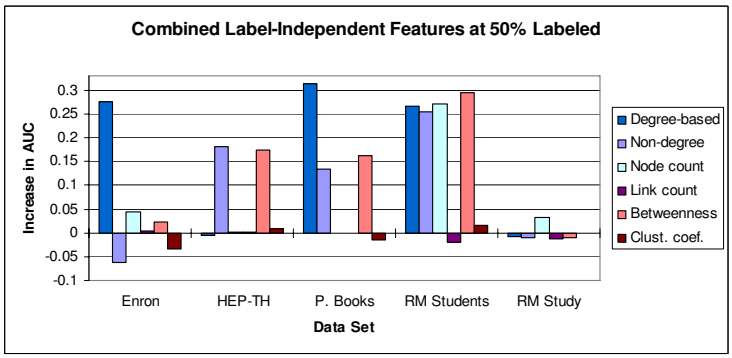

Fig. 6. Performance of LI features in combination. Degree-based features are node and link count. Non-degree features are betweenness and clustering coefficient.

For most tasks, performance improves due to using all four LI features. On Enron, however, clustering coefficient appears to mislead the classifier to the point where it is better to use either node or link count individually than to use all features. This is one case where we might benefit from a more selective classifier. Figure 7 compares logistic regression with a random forest classifier [25], both using the same four LI features. As expected, the random forest is better able to make use of the informative features without being misled by the uninformative ones.

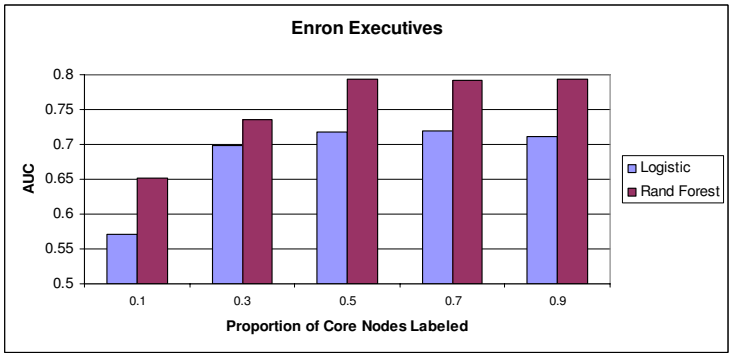

Fig. 7. Comparison of logistic regression and random forest classifiers with all four LI features.

To get a feel for why some LI features make better predictors than others, we examine the distribution of each feature by class for each prediction task. Table 2

summarizes these feature distributions by their mean and standard deviation. In general, we expect features that cleanly separate the classes to provide the most predictive power. As mentioned previously, clustering coefficient appears to be the least powerful feature overall for our set of prediction tasks. One possible explanation for clustering coefficient's general poor performance is that it does not vary enough from node to node; therefore, it does not help to differentiate among instances of different classes.

Table 2. Mean and standard deviation (SD) of feature values by class and data set. The larger mean value for each feature (i.e., row) is shown in bold.

Data Set/Feature	**Mean (SD) for the '+' Class**	**Mean (SD) for the '-' Class**
Political Books	Neutral	Other
Node Count	5.8 (3.3)	**8.8** (5.6)
Link Count	5.8 (3.3)	**8.8** (5.6)
Betweenness	**0.027** (0.030)	0.019 (0.029)
Clust. Coef.	0.486 (0.25)	**0.489** (0.21)
Enron	Executive	Other
Node Count	**22** (27)	9.6 (20)
Link Count	**61** (100)	25 (66)
Betweenness	**0.0013** (0.0037)	0.00069 (0.0025)
Clust. Coef.	0.91 (0.77)	**1.75** (4.5)
RM Student	Student	Other
Node Count	19 (27)	**22** (38)
Link Count	471 (774)	**509** (745)
Betweenness	**0.027** (0.050)	0.022 (0.056)
Clust. Coef.	**15** (22)	8.0 (7.0)
RM Study	In-study	Out-of-study
Node Count	**18** (30)	1.4 (2.8)
Link Count	**418** (711)	30 (130)
Betweenness	**0.022** (0.048)	0.00086 (0.022)
Clust. Coef.	**10** (17)	5.8 (51)
HEP-TH	Differential Geometry	Other
Node Count	14 (9.0)	**21** (26)
Link Count	14 (9.0)	**21** (26)
Betweenness	0.000078 (0.00010)	**0.0011** (0.0056)
Clust. Coef.	**0.42** (0.19)	0.40 (0.23)

Figure 8 shows the degree of variability of each LI feature across the five prediction tasks. To measure variability, we use the *coefficient of variation*, a normalized measure of the dispersion of a probability distribution. The coefficient of variation is defined as:

$$cv(dist) = \frac{\sigma}{\mu} \tag{4}$$

where μ is the mean of the probability distribution $dist$ and σ is the standard deviation. A higher coefficient of variation indicates a feature with more varied values across instances in the data set.

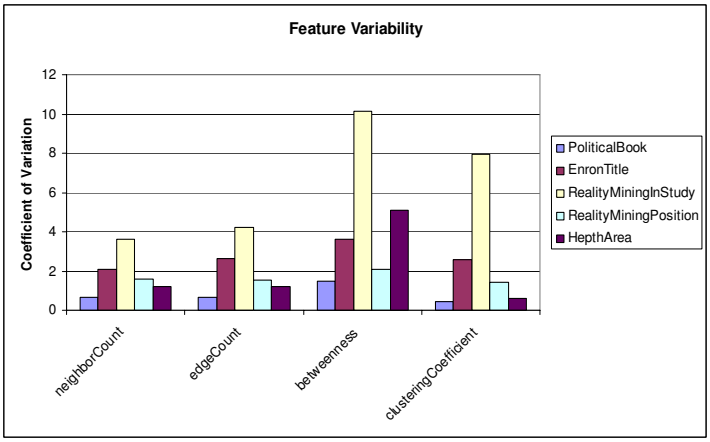

Fig. 8. Degree of variability for each LI feature on each prediction task.

The variability of the clustering coefficient appears comparable to the degree features (i.e., node and link count) (see Figure 8). We even observe that the degree of variability of the clustering coefficient for the Enron task is higher than the degree of variability for the neighbor count feature, even though neighbor count provides much more predictive power (see Figure 5). So, clustering coefficient appears to have sufficient variability over the nodes in the graph. However, it is possible that the clustering coefficient exhibits similar variability for nodes of both classes; and thus, still fails to adequately distinguish between nodes of different classes. Therefore, we wish to quantify the extent to which the feature distributions can be separated from one another by class.

Figure 9 shows how well each LI feature separates the two classes for each prediction task. We measure class separation by calculating the distance between the empirical distributions of the LI feature values for each class. Specifically, we use the *Kolmogorov-Smirnov statistic* (K-S) to measure the distance between two empirical (cumulative) distribution functions:

$$D(F_n(x), G_n(x)) = max(|F_n(x) - G_n(x)|) \tag{5}$$

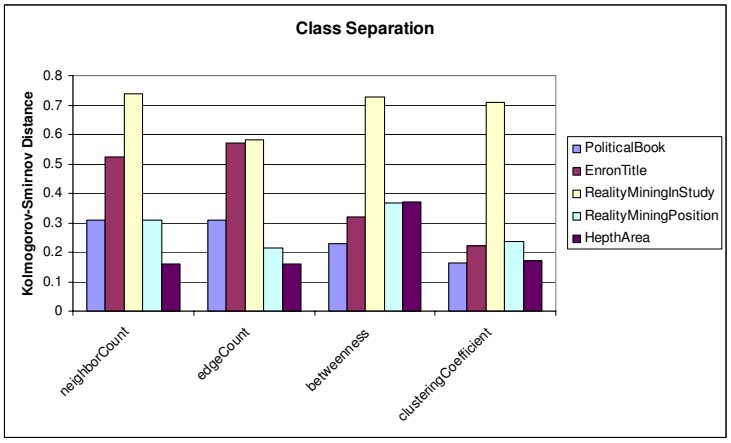

Fig. 9. Degree of class separation for each LI feature on each prediction task.

We observe from Figure 9 that, on average, the per-class distributions of values for the clustering coefficient are more similar to one another than for other LI features. So, although the clustering coefficient does vary from node to node, the values do not differ consistently based on class. Therefore, clustering coefficient has a hard time distinguishing between instances of different classes, and exhibits poor predictive power overall. The exception is on the Reality Mining study-participant task, where we observe a high K-S distance (Figure 9) and a correspondingly high classification performance (Figure 5). In fact, the K-S distances in Figure 9 generally correspond quite well to the classification performance we observe in Figure 5.

5.4 Observations about Our Problem Domains and Data Sets

Table 2 highlights a number of interesting characteristics of our data sets. An examination of these characteristics provide insights into the underlying problem domains. We describe several such insights here.

More politically extreme books tend to have higher degree (neighboring nodes and adjacent edges) and clustering coefficient, but lower betweenness than the neutral books. This tells us that there are two very different types of readers represented in our network: (1) party loyalists that tend to have more extreme viewpoints, strong agreement with others inside their party, and strong disagreement with outsiders and (2) political moderates who are more inclined to consider multiple differing perspectives on an issue.

Enron executives tend to have higher degree and betweenness, but lower clustering coefficients than others. So, as we would expect, executives maintain more relationships than other employees and are positioned in a place of maximal control over information flow. The lower clustering coefficient suggests that

executives maintain ties to multiple communities within the company and are less closely tied to a particular community.

Reality Mining students tend to have higher betweenness and clustering coefficient, but lower degree than others. This indicates that students tend to be more cliquey, with many of their communications made within a single strong community. It also indicates that students play an important role in keeping information flowing between more distant parts of the network.

Reality Mining study participants tend to have higher degree, betweenness, and clustering coefficient than non-participants. These findings may reveal more about how the data were collected than about the underlying problem domain, but they are interesting nonetheless. Because their phones were instrumented with special software, we have information on all calls of study participants. However, we have only a partial view of the calls made and received by non-participants. More specifically, the only calls we observe for non-participants are those to or from a study participant. The result of this is that we end up with a central community of interconnected study participants, surrounded by a large, but diffuse periphery of non-participants. Thus, the participants appear to have more neighbors, higher centrality, and a more closely knit surrounding community.

In HEP-TH, differential geometry papers tend to have higher clustering coefficient, but lower degree and betweenness than others topics. This indicates that differential geometry papers play a relatively isolated and peripheral role among high-energy physics papers, at least in our subset of the arXiv data.

6 Conclusion

We examined the utility of label-independent features in the context of within-network classification. Our experiments revealed a number of interesting findings: (1) LI features can make up for large amounts of missing class labels; (2) LI features can provide information above and beyond that provided by class labels alone; (3) the effectiveness of LI features is due, at least in part, to their consistency and their stabilizing effect on network classifiers; (4) no single label-independent feature dominates, and there is generally a benefit to combining a few diverse LI features. In addition, we observed a benefit to combining LI features with label propagation, although the benefit is not consistent across tasks.

Our findings suggest a number of interesting areas for future work. These include:

- Combining attribute-based (LD) and structural-based (LI) features of a network to create new informative features for node classification. For instance, will the number of short paths to nodes of a certain label or the average path length to such nodes improve classification performance?
- Exploring the relationship between attributes and network structure in time-evolving networks, where links appear and disappear and attribute values change over time. For example, in such a dynamic network, could we use

a time-series of LI feature values to predict the values of class labels at a future point in time?

7 Acknowledgments

We would like to thank Luke McDowell for his insightful comments. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract No. W-7405-ENG-48 and No. DE-AC52-07NA27344 (LLNL-JRNL-408814).

References

1. Taskar, B., Abbeel, P., Koller, D.: Discriminative probabilistic models for relational data. In: Proceedings of the 18th Conference on Uncertainty in AI. (2002) 485–492
2. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. AI Magazine **29**(3) (2008) 93–106
3. Neville, J., Jensen, D.: Relational dependency networks. Journal of Machine Learning Research **8** (2007) 653–692
4. Getoor, L., Friedman, N., Koller, D., Taskar, B.: Learning probabilistic models of link structure. Journal of Machine Learning Research **3** (2002) 679–707
5. Lu, Q., Getoor, L.: Link-based classification. In: Proceedings of the 20th International Conference on Machine Learning. (2003) 496–503
6. Neville, J., Jensen, D., Gallagher, B.: Simple estimators for relational bayesian classifiers. In: Proceedings the 3rd IEEE International Conference on Data Mining. (2003) 609–612
7. Macskassy, S., Provost, F.: Classification in networked data: A toolkit and a univariate case study. Journal of Machine Learning Research **8** (2007) 935–983
8. Neville, J., Jensen, D., Friedland, L., Hay, M.: Learning relational probability trees. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. (2003) 625–630
9. Perlich, C., Provost, F.: Aggregation-based feature invention and relational concept classes. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. (2003) 167–176
10. Singh, L., Getoor, L., Licamele, L.: Pruning social networks using structural properties and descriptive attributes. In: Proceedings of the 5th IEEE International Conference on Data Mining. (2005) 773–776
11. Neville, J., Jensen, D.: Leveraging relational autocorrelation with latent group models. In: Proceedings the 5th IEEE International Conference on Data Mining. (2005) 322–329
12. Chakrabarti, S., Dom, B., Indyk, P.: Enhanced hypertext categorization using hyperlinks. In: Proceedings of ACM SIGMOD International Conference on Management of Data. (1998) 307–318
13. Jensen, D., Neville, J., Gallagher, B.: Why collective inference improves relational classification. In: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. (2004) 593–598
14. McDowell, L., Gupta, K., Aha, D.: Cautious inference in collective classification. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence. (2007) 596–601

15. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using gaussian fields and harmonic functions. In: Proceedings of the 20th International Conference on Machine Learning. (2003) 912–919
16. Zhu, X.: Semi-supervised learning literature survey. Technical Report CS-TR-1530, University of Wisconsin, Madison, WI (December 2007) http://pages.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf.
17. Gallagher, B., Tong, H., Eliassi-Rad, T., Faloutsos, C.: Using ghost edges for classification in sparsely labeled networks. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. (2008) 256–264
18. Newman, M.: The structure and function of complex networks. SIAM Review **45** (2003) 167–256
19. Macskassy, S., Provost, F.: A simple relational classifier. In: Notes of the 2nd Workshop on Multi-relational Data Mining at KDD'03. (2003)
20. Gallagher, B., Eliassi-Rad, T.: An examination of experimental methodology for classifiers of relational data. In: Proceedngs of the 7th IEEE International Conference on Data Mining Workshops. (2007) 411–416
21. Krebs, V.: Books about U.S. politics. http://www.orgnet.com/divided2.html (2004)
22. Cohen, W.: Enron email data set. http://www.cs.cmu.edu/~enron/
23. Eagle, N., Pentland, A.: Reality mining: sensing complex social systems. Journal of Personal and Ubiquitous Computing **10**(4) (2006) 255–268 Data available at http://reality.media.mit.edu.
24. Jensen, D.: Proximity HEP-TH database. http://kdl.cs.umass.edu/data/hepth/hepth-info.html
25. Breiman, L.: Random forests. Machine Learning **45**(1) (2001) 5–32